

AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A system for detecting termination of an application instance using locks, comprising:
 - a holding child process ~~forked from a parent process, the holding child process comprising a connection to a monitored application instance and configured to obtain~~ [[an]] a first exclusive lock on an object maintained by the monitored application instance, the holding child process returning a ready signal upon successfully acquiring the exclusive lock;
 - a waiting child process ~~forked from the parent process subsequent to the holding child process, the waiting child process comprising a connection to the monitored application instance, the waiting child process configured to (a) request a second exclusive lock on the object after the holding process has been granted the first exclusive lock on the object, blocking on the exclusive lock on the monitored application instance and returning and (b) return a result signal, to a monitor process, upon at least one of acquiring the second exclusive lock and ceasing to be blocked clearing the block on the exclusive lock; and~~
 - the ~~monitor parent~~ process configured to process processing the result return signal to determine whether the application instance has terminated.
2. (Currently amended) A system according to Claim 1, further comprising:
 - the ~~monitor parent~~ process determining whether the application instance terminated based, at least in part, on whether the monitor process receives processing a standard error or a non-standard error received from the waiting child process.

3. (Currently amended) A system according to Claim 1, wherein the holding process resides at the same node as the application instance, and where the waiting process does not reside at the same node as the application instance ~~further comprising:~~
~~the parent process processing a non-standard error received from the waiting child process.~~
4. (Currently amended) A system according to Claim 3, further comprising:
a validation module configured to (a) check ~~checking~~ for termination of the monitored application and (b) signal ~~signaling~~ termination of the monitored application to a cluster service.
5. (Currently amended) A system according to Claim 3, further comprising:
a validation module configured to (a) check ~~checking~~ for termination of the monitored application and (b) restart ~~restarting~~ the holding ~~child~~ process and the waiting ~~child~~ process.
6. (Currently amended) A system according to Claim 1, wherein the application instance is ~~comprises~~ a database service instance.
7. (Currently amended) A method for detecting termination of an application instance using locks, comprising:
starting a holding ~~child~~ process configured to perform the steps of ~~from a parent process, comprising:~~
~~connecting to a monitored application instance;~~
(a) acquiring [[an]] a first exclusive lock on an object maintained by the
~~monitored application instance,[[;]] and~~
(b) returning a ready signal, to a monitor process, upon successfully acquiring
the first exclusive lock; and
in response to receiving the ready signal, starting a waiting child process configured to
perform the steps of ~~from the parent process subsequent to the holding child process, comprising:~~

(a) connecting to the ~~monitored~~ application instance,[[:]]
 (b) ~~blocking~~ requesting a second ~~on the~~ exclusive lock on the object maintained by the monitored application instance,[[:]] and
 (c) returning, to the monitor process, a result signal upon at least one of acquiring the second exclusive lock and ceasing to be blocked ~~clearing the block on the exclusive lock;~~ and
 processing the result return signal, at the ~~parent~~ monitor process, to determine whether the application instance has terminated.

8. (Currently amended) A method according to Claim 7, further comprising:
determining whether the application instance terminated based, at least in part, on whether the monitor process receives ~~processing~~ a standard error or non-standard error received from the waiting ~~child~~ process.
9. (Currently amended) A method according to Claim 7, wherein the holding process resides at the same node as the application instance, and wherein the waiting process does not reside at the same node as the application instance ~~further comprising:~~
~~processing a non-standard error received from the waiting child process.~~
10. (Currently amended) A method according to Claim 9, further comprising the steps of:
 checking for termination of the ~~monitored~~ application instance; and
 signaling termination of the ~~monitored~~ application instance to a cluster service.
11. (Currently amended) A method according to Claim 9, further comprising the steps of:
 checking for termination of the monitored application; and
 restarting the holding ~~child~~ process and the waiting ~~child~~ process.
12. (Currently amended) A method according to Claim 7, wherein the application instance ~~comprises~~ is a database server instance.

13. (Currently amended) A machine-readable medium carrying one or more sequences of instructions, which when executed by one or more processors, causes the one or more processors to perform the steps of method recited in Claim 7.
~~A computer readable storage medium holding code for performing the method according to Claim 7.~~
14. (Currently amended) A system for detecting termination of a database instance using events, comprising:
a waiting process subroutine ~~forked from a main routine, the waiting subroutine comprising a connection to a monitored database instance and blocking configured to block on a named event in the database instance and return~~
returning a result to the a main routine upon an occurrence of the named event;
and
the main process routine configured to process ~~processing~~ the result to determine whether the database instance has terminated.
15. (Currently amended) A system according to Claim 14, wherein the named event comprises a membership change event, and the method further comprises ~~comprising:~~
at the main process routine, processing the membership change event and restarting the waiting process subroutine.
16. (Currently amended) A system according to Claim 14, wherein further comprising:
the main process routine determining whether the database instance terminated based, at least in part, on whether the monitor process receives ~~processing~~ a standard error or a non-standard error received from the waiting process subroutine.
17. (Currently amended) A system according to Claim 14, wherein the waiting process resides at a different node than the database instance ~~further comprising:~~
~~the main routine processing a non-standard error received from the waiting subroutine.~~
18. (Currently amended) A system according to Claim 17, further comprising:

a validation module configured to (a) check ~~checking~~ for termination of the database instance ~~monitored named instance application~~ and (b) ~~signal~~ signaling termination of the database instance ~~monitored named instance application~~ to a cluster service.

19. (Currently amended) A system according to Claim 17, further comprising:
a validation module configured to (a) check ~~checking~~ for termination of the database instance ~~monitored named instance application~~ and (b) ~~restart~~ restarting the waiting process ~~subroutine~~.
20. (Currently amended) A method for detecting termination of a database instance ~~using events~~, comprising:
starting a waiting process ~~subroutine from a main routine, comprising~~ configured to perform the steps of:
 - (a) connecting to the ~~a monitored~~ database instance;
 - (b) waiting for ~~blocking on~~ a named event ~~to occur~~ in the database instance; and
 - (c) returning a result to the ~~main a routine~~ ~~main process~~ upon an occurrence of the named event; and
 - (d) processing the result at the main process ~~routine to determine whether the database instance has terminated~~.
21. (Currently amended) A method according to Claim 20, wherein the named event comprises a membership change event, further comprising:
processing the membership change event at the main ~~process routine~~; and
restarting the waiting process ~~subroutine~~.
22. (Currently amended) A method according to Claim 20, further comprising:
processing a standard error or a non-standard error received from the waiting process ~~subroutine~~.

23. (Currently amended) A method according to Claim 20, wherein the waiting process resides at a different node than the database instance ~~further comprising:~~
~~processing a non-standard error received from the waiting subroutine.~~
24. (Currently amended) A method according to Claim 23, further comprising:
checking for termination of the database instance ~~monitored named instance application;~~
and
signaling termination of the database instance ~~monitored named instance application~~ to
a cluster service.
25. (Currently amended) A method according to Claim 23, further comprising:
checking for termination of the database instance ~~monitored named instance application;~~
and
restarting the waiting process ~~subroutine.~~
26. (Currently amended) ~~A computer-readable storage medium holding code for~~
~~performing the method according to Claim 20~~ A machine-readable medium carrying one
or more sequences of instructions, which when executed by one or more processors,
causes the one or more processors to perform the steps of method recited in Claim 20.